

## References

- [1] Gabriele Cesa, Leon Lang, and Maurice Weiler. A program to build  $E(N)$ -equivariant steerable CNNs. In *International Conference on Learning Representations*, 2022. URL <https://openreview.net/forum?id=WE4qe9xlnQw>
- [2] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Proceedings of Robotics: Science and Systems (RSS)*, 2023.
- [3] MMDetection3D Contributors. MMDetection3D: OpenMMLab next-generation platform for general 3D object detection. <https://github.com/open-mmlab/mmdetection3d>, 2020.
- [4] Theophile Gervet, Zhou Xian, Nikolaos Gkanatsios, and Katerina Fragkiadaki. Act3d: Infinite resolution action detection transformer for robotic manipulation. *arXiv preprint arXiv:2306.17817*, 2023.
- [5] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising Diffusion Probabilistic Models. *Advances in Neural Information Processing Systems*, 33:6840–6851, 2020.
- [6] Stephen James and Andrew J Davison. Q-attention: Enabling efficient learning for vision-based robotic manipulation. *IEEE Robotics and Automation Letters*, 7(2):1612–1619, 2022.
- [7] Stephen James, Marc Freese, and Andrew J. Davison. Pyrep: Bringing v-rep to deep robot learning. *arXiv preprint arXiv:1906.11176*, 2019.
- [8] Stephen James, Zicong Ma, David Rovick Arrojo, and Andrew J. Davison. Rlbench: The robot learning benchmark & learning environment. *IEEE Robotics and Automation Letters*, 5(2): 3019–3026, 2020. doi: 10.1109/LRA.2020.2974707.
- [9] Stephen James, Kentaro Wada, Tristan Laidlow, and Andrew J Davison. Coarse-to-fine q-attention: Efficient learning for visual robotic manipulation via discretisation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13739–13748, 2022.
- [10] Leonid Keselman, John Iselin Woodfill, Anders Grunnet-Jepsen, and Achintya Bhowmik. Intel realsense stereoscopic depth cameras. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 1–10, 2017.
- [11] Andrew Levy, Robert Platt, and Kate Saenko. Hierarchical reinforcement learning with hindsight. *arXiv preprint arXiv:1805.08180*, 2018.
- [12] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL <https://arxiv.org/abs/1711.05101>
- [13] Xiao Ma, Sumit Patidar, Iain Haughton, and Stephen James. Hierarchical diffusion policy for kinematics-aware multi-task robotic manipulation. *CVPR*, 2024.
- [14] E. Rohmer, S. P. N. Singh, and M. Freese. V-rep: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 1321–1326, 2013. doi: 10.1109/IROS.2013.6696520.
- [15] Hyunwoo Ryu, Hong in Lee, Jeong-Hoon Lee, and Jongeun Choi. Equivariant Descriptor Fields:  $SE(3)$ -Equivariant Energy-Based Models for End-to-End Visual Robotic Manipulation Learning. In *The Eleventh International Conference on Learning Representations*, 2023.
- [16] Sahil Sharma, Aravind Suresh, Rahul Ramesh, and Balaraman Ravindran. Learning to factor policies and action-value functions: Factored action space representations for deep reinforcement learning. *arXiv preprint arXiv:1705.07269*, 2017.
- [17] Mohit Shridhar, Lucas Manuelli, and Dieter Fox. Perceiver-actor: A multi-task transformer for robotic manipulation. In *Conference on Robot Learning*, pages 785–799. PMLR, 2023.

- 158 [18] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*,  
159 2017.
- 160 [19] Dian Wang, Colin Kohler, and Robert Platt. Policy learning in se (3) action spaces. In  
161 *Proceedings of the Conference on Robot Learning*, 2020.
- 162 [20] Dian Wang, Stephen Hart, David Surovik, Tarik Kelestemur, Haojie Huang, Haibo Zhao, Mark  
163 Yeatman, Jiuguang Wang, Robin Walters, and Robert Platt. Equivariant diffusion policy. *arXiv*  
164 *preprint arXiv:2407.01812*, 2024.
- 165 [21] Zhou Xian, Nikolaos Gkanatsios, Theophile Gervet, Tsung-Wei Ke, and Katerina Fragkiadaki.  
166 Chaineddiffuser: Unifying trajectory diffusion and keypose prediction for robotic manipulation.  
167 In *7th Annual Conference on Robot Learning*, 2023.
- 168 [22] Tony Z Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual  
169 manipulation with low-cost hardware. *arXiv preprint arXiv:2304.13705*, 2023.
- 170 [23] Yin Zhou and Oncel Tuzel. Voxelnet: End-to-end learning for point cloud based 3d object  
171 detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*,  
172 pages 4490–4499, 2018.

## 173 **A Proof: The Full Policy is $SO(2)$ Equivariant**

174 Let us prove that the policy is  $SO(2)$  Equivariant and satisfies

$$\forall g \in SO(2), \quad \pi(g \cdot o) = g \cdot \pi(o) \quad (1)$$

175 We will prove this in two steps.

### 176 **A.1 Low-level Equivariance**

177 First, let us prove that the low-level agent is  $SO(2)$  equivariant. The low-level policy can be written  
178 as

$$\pi_{low}(o, t_{high}) = \tau(\phi(\tau(o, t_{high})), -t_{high})$$

179 The frame transfer functions satisfy

$$\forall g \in SO(2), \quad \tau(g \cdot o, g \cdot t) = g \cdot \tau(o, t)$$

180 and the diffusion policy satisfies

$$\forall g \in SO(2), \quad \phi(g \cdot o) = g \cdot \phi(o)$$

181 Thus, we have that

$$\forall g \in SO(2), \pi_{low}(g \cdot o, g \cdot t_{high}) = \tau(\phi(\tau(g \cdot o, g \cdot t_{high})), g \cdot -t_{high})$$

182 Using the frame transfer function property  $\tau(g \cdot o, g \cdot t) = g \cdot \tau(o, t)$  we have that

$$\forall g \in SO(2), \pi_{low}(g \cdot o, g \cdot t_{high}) = \tau(\phi(g \cdot \tau(o, t_{high})), g \cdot -t_{high})$$

183 Using the  $SO(2)$  equivariance of the diffusion policy and the properties of the frame transfer functions,  
184 we have that

$$\forall g \in SO(2), \tau(\phi(g \cdot \tau(o, t_{high})), g \cdot -t_{high}) = \tau(g \cdot \phi(\tau(o, t_{high})), g \cdot -t_{high}) = g \cdot \tau(\phi(\tau(o, t_{high})), -t_{high})$$

185 Thus,

$$\forall g \in SO(2), \pi_{low}(g \cdot o, g \cdot t_{high}) = g \cdot \tau(\phi(\tau(o, t_{high})), -t_{high})$$

186 And by definition,

$$\tau(\phi(\tau(o, t_{high})), -t_{high}) = \pi_{low}(o, t_{high})$$

187 Thus, we have that

$$\forall g \in SO(2), \pi_{low}(g \cdot o, g \cdot t_{high}) = g \cdot \pi_{low}(o, t_{high})$$

188

□

### 189 **A.2 Full Policy Equivariance**

190 Using the equivariance of the low-level policy, let us show that the full policy is  $SO(2)$  equivariant.

191 The high-level, low-level and diffusion policies satisfy

$$\begin{aligned} \forall g \in SO(2), \quad \pi_{high}(g \cdot o) &= g \cdot \pi_{high}(o) \\ \forall g \in SO(2), \quad \pi_{low}(g \cdot o, g \cdot t_{high}) &= g \cdot \pi_{low}(o, t_{high}) \end{aligned}$$

192 Now, combining high-level and low-level policy together we got that:

$$\pi(o) = \pi_{low}(\pi_{high}(o), o)$$

193 When  $g$  acting on both the input observation, we have that

$$\forall g \in SO(2), \quad \pi(g \cdot o) = \pi_{low}(\pi_{high}(g \cdot o), g \cdot o)$$

194 Now, via the  $SO(2)$  equivariance of the high-level policy  $\pi_{high}(g \cdot o) = g \cdot \pi_{high}(o)$  we have that

$$\forall g \in SO(2), \pi_{low}(\pi_{high}(g \cdot o), g \cdot o) = \pi_{low}(g \cdot \pi_{high}(o), g \cdot o)$$

195 Thus, using the  $SO(2)$  equivariance of the low-level policy  $\pi_{low}(g \cdot \pi_{high}(o), g \cdot o) = g \cdot$   
196  $\pi_{low}(\pi_{high}(o), o)$  we have that

$$\forall g \in SO(2), \quad \pi(g \cdot o) = g \cdot \pi_{low}(\pi_{high}(o), o)$$

197 Note that the  $\pi_{low}(\pi_{high}(o), o)$  is just the expression for  $\pi(o)$ . Thus, we must have that

$$\forall g \in SO(2), \quad \pi(g \cdot o) = g \cdot \pi(o)$$

198 holds.

□

## B Proof: The Full Policy is $T(3)$ Equivariant

As defined in [subsection 4.1](#),  $+$ ,  $-$  as operators between  $o$  or  $a$  and  $t_{\text{high}}$  as addition and subtraction on the  $(x, y, z)$  component of  $o$  or  $a$ . Similarly, we can define the translation  $t \in T(3)$  acting on  $o$  or  $a$  as an addition to the  $(x, y, z)$  component as  $o + t$  or  $a + t$ .

First, suppose that the high-level policy  $\pi_{\text{high}}(o)$  is  $T(3)$ -equivariant so that

$$\forall t \in T(3), \pi_{\text{high}}(o + t) = t + \pi_{\text{high}}(o)$$

which is simply the statement that shifting the scene shifts the high-level policy in the same way. Now, we will show that the full hierarchical policy satisfies the equivariance condition. The low-level policy is determined by

$$\pi_{\text{low}}(\pi_{\text{high}}(o), o) = \tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o))$$

How does the low-level policy transform under a translation? Using  $\pi(o) = \pi_{\text{low}}(\pi_{\text{high}}(o), o)$ , we have that

$$\pi(o + t) = \pi_{\text{low}}(o + t, \pi_{\text{high}}(o + t))$$

Using the definition of the low-level policy, we have that

$$\pi(o + t) = \pi_{\text{low}}(o + t, \pi_{\text{high}}(o + t)) = \tau(\phi(\tau(o + t, \pi_{\text{high}}(o + t))), -\pi_{\text{high}}(o + t))$$

Now, using the equivariance of high-level policy, we have that  $\pi_{\text{high}}(o + t) = t + \pi_{\text{high}}(o)$  so that

$$\tau(\phi(\tau(o + t, \pi_{\text{high}}(o + t))), -\pi_{\text{high}}(o + t)) = \tau(\phi(\tau(o + t, \pi_{\text{high}}(o) + t)), -\pi_{\text{high}}(o) - t)$$

Now, we can simplify this expression via the fact that the frame transfer  $\tau$  function is  $T(3)$  invariant.

We have that  $\tau(o + t, \pi_{\text{high}}(o) + t) = \tau(o, t_{\text{high}})$  which implies that

$$\tau(\phi(\tau(o + t, \pi_{\text{high}}(o) + t)), -\pi_{\text{high}}(o) - t) = \tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o) - t)$$

Now, note that  $\tau(o + t, \pi_{\text{high}}(o) + t) = \tau(o, t_{\text{high}})$  implies that  $\tau(o, -\pi_{\text{high}}(o) - t) = \tau(o + t, -\pi_{\text{high}}(o))$ . Using the fact that  $\tau(o + t, -\pi_{\text{high}}(o)) = \tau(o, -\pi_{\text{high}}(o)) + t$ , we have that

$$\tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o) - t) = \tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o)) + t$$

Thus, combining the above expressions and using the definition of the low-level policy, we have that

$$\pi_{\text{low}}(o + t, \pi_{\text{high}}(o + t)) = \tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o) - t) = t + \tau(\phi(\tau(o, \pi_{\text{high}}(o))), -\pi_{\text{high}}(o)) = t + \pi_{\text{low}}(o, \pi_{\text{high}}(o))$$

Thus, we have that

$$\pi_{\text{low}}(o + t, \pi_{\text{high}}(o + t)) = t + \pi_{\text{low}}(o, \pi_{\text{high}}(o))$$

This is just the expression for the full policy function as  $\pi(o) = \pi_{\text{low}}(o, \pi_{\text{high}}(o))$ . Ergo, we must have that

$$\pi(o + t) = t + \pi(o)$$

and the full policy is  $T(3)$ -Equivariant.  $\square$

## C Proof: Equivariance of the Stacked Voxel Representation

*Proof.* We aim to prove that the stacked voxel representation  $\nu$  is  $T(3) \times SO(2)$ -equivariant, i.e.,

$$\nu(gP) = g\nu(P),$$

where  $g \in T(3) \times SO(2)$  is a group transformation.

Define a point set selection function  $m : j_x, j_y, j_z, P \mapsto P_j$  that selects the subset of points  $P_j \subseteq P$  within the voxel indexed by  $(j_x, j_y, j_z)$ . By the definition,  $m$  is an equivariant function  $m(g(j_x, j_y, j_z), gP) = gm(j_x, j_y, j_z, P)$ .

The stacked voxel representation  $\nu$  for a given voxel location  $j_x, j_y, j_z$  can be written as:

$$\nu(P)(j_x, j_y, j_z) = l(m(j_x, j_y, j_z, P)),$$

where:



- $m(j_x, j_y, j_z, P)$  selects the subset of points  $P_j \subseteq P$  within the voxel indexed by  $(j_x, j_y, j_z)$ ,
- $l(P_j)$  maps the selected point subset  $P_j$  to a feature vector representing the voxel at  $(j_x, j_y, j_z)$ .

Substitute  $P = gP$  into  $\nu(P)$ . Using the definition, we have:

$$\nu(gP)(j_x, j_y, j_z) = l(m(j_x, j_y, j_z, gP)).$$

The point set selection function  $m$  is equivariant to group transformations, so we have:

$$m(j_x, j_y, j_z, gP) = g m(g^{-1}(j_x, j_y, j_z), P).$$

Substitute this into the expression for  $\nu(gP)$ :

$$\nu(gP)(j_x, j_y, j_z) = l(g m(g^{-1}(j_x, j_y, j_z), P)).$$

The PointNet  $l$  is  $SO(2)$ -equivariant and  $T(3)$ -invariant, meaning:

$$l(gP_j) = \rho(\theta)l(P_j),$$

where  $P_j = m(j_x, j_y, j_z, P)$ , and  $\rho(\theta)$  is the linear representation of the rotation group action.

Applying this to  $l(g m(g^{-1}(j_x, j_y, j_z), P))$ :

$$\nu(gP)(j_x, j_y, j_z) = \rho(\theta)l(m(g^{-1}(j_x, j_y, j_z), P)).$$

From the definition of  $\nu(P)$ , we know:

$$l(m(g^{-1}(j_x, j_y, j_z), P)) = \nu(P)(g^{-1}(j_x, j_y, j_z)).$$

Substituting this into the equation:

$$\nu(gP)(j_x, j_y, j_z) = \rho(g)\nu(P)(g^{-1}(j_x, j_y, j_z)).$$

Since  $\mathcal{V} = \nu(P)$  is a voxel grid (in function representation), the group action on  $\nu$  is defined as:

$$(g\nu(P))(j_x, j_y, j_z) = \rho(g)\nu(P)(g^{-1}(j_x, j_y, j_z)).$$

Thus, we have:

$$\nu(gP) = g\nu(P).$$

□

## D Additional Background of Group Symmetry

The group  $T(3) \times SO(2)$  can be naturally decomposed into translation  $T(3)$ , which is handled using methods like 3D convolution, and rotation  $SO(2)$ , which is addressed via network design by introducing equivariant layers [1] that respect  $SO(2)$  transformations through appropriate representations of  $SO(2)$  or its subgroups.

### D.1 Group Action of $SO(2)$

We focus on three particular representations of  $g \in SO(2)$  or its subgroup  $g \in C_u$  (containing  $u$  discrete rotations) that define how the group acts on different data. Specifically:

Trivial Representation  $\rho_0$ : The trivial representation  $\rho_0$  characterizes the action of  $SO(2)$  or  $C_u$  on an invariant scalar  $x \in \mathbb{R}$  such that  $\rho_0(g)x = x$ . This means that every group element  $g$  leaves the scalar  $x$  unchanged.

Standard Representation  $\rho_1$ : The standard representation  $\rho_1$  defines how  $SO(2)$  or  $C_u$  acts on a vector  $v \in \mathbb{R}^2$  using a  $2 \times 2$  rotation matrix. The action is given by  $\rho_\omega(g)v = \begin{pmatrix} \cos g & -\sin g \\ \sin g & \cos g \end{pmatrix} v$ . When  $\omega = 1$ , the representation  $\rho_1(g)$  corresponds to the standard  $2 \times 2$  rotation matrix.

256 **Regular Representation  $\rho_{\text{reg}}$ :** The regular representation  $\rho_{\text{reg}}$  describes the action of  $C_u$  on a  
257 vector  $x \in \mathbb{R}^u$  via  $u \times u$  permutation matrices. Let  $g = r^m$  be an element of the cyclic group  
258  $C_u = \{1, r^1, \dots, r^{u-1}\}$ , and let  $x = (x_1, x_2, \dots, x_u) \in \mathbb{R}^u$ . Then the action is defined by  
259  $\rho_{\text{reg}}(g)x = (x_{u-m+1}, x_{u-m+2}, \dots, x_u, x_1, x_2, \dots, x_{u-m})$ . This operation cyclically permutes the  
260 coordinates of  $x$  in  $\mathbb{R}^u$ .

261  
262 A representation  $\rho$  can also be constructed as a combination of different representations.  
263 Specifically,  $\rho$  is defined as the direct sum  $\rho = \rho_0^{n_0} \oplus \rho_1^{n_1} \oplus \rho_2^{n_2}$ , which belongs to the gen-  
264 eral linear group  $GL(n_0 + 2n_1 + 2n_2)$ . In this case,  $\rho(g)$  is a block diagonal matrix of size  
265  $(n_0 + 2n_1 + 2n_2) \times (n_0 + 2n_1 + 2n_2)$  that acts on vectors  $x \in \mathbb{R}^{n_0+2n_1+2n_2}$ .

## 266 D.2 Group Action of $T(3)$

267 Follow the definition of + - The group  $T(3)$  of 3D translations is an additive group, whose action  
268 is defined by shifting spatial coordinates. For example, for a point cloud  $P = \{p_1, p_2, \dots\}$  where  
269  $p_i = (x_i, y_i, z_i)$ , the action of  $g \in T(3)$  is  $t \cdot p_i = (x_i + t_x, y_i + t_y, z_i + t_z)$ . Similarly, for voxel-based  
270 representations,  $T(3)$  acts by shifting the spatial indices of the voxel grid. Convolutions are inherently  
271 translationally invariant. - I would say this Translation symmetry is naturally handled by operations  
272 such as 3D convolutions, which are inherently translation-equivariant.

## 273 E Training Detail

274 In the simulation experiments, we use a batch size of 16 for training. Specifically, the observation  
275 contains one step of history observation, and 3 steps of history action and the output of the denoising  
276 process is a sequence of 18 action steps. In close-loop control we use all 18 steps for training and  
277 execute 18 steps, similar to prior work ([21]). In close-loop control 18 steps and 9 steps are used  
278 for training and execution, similar to setting of [20] a. We train our models with the AdamW ([12])  
279 optimizer (with a learning rate of  $10^{-4}$  and weight decay of  $5 \cdot 10^{-4}$ ). We use DDPM ([5]) with 100  
280 denoising steps for both training and evaluation. We training each tasks with 100000 iterates.

## 281 F Detail of Simulation Tasks

282 Here are descriptions of 30 tasks, as shown in Figure 3 mentioned in simulation experiment:

- 283 1. **Pick/Lift:** Grasp and lift a block from the table.
- 284 2. **Push Button:** Press a button.
- 285 3. **Knife on Board:** Place a knife onto a cutting board.
- 286 4. **Put Money:** Put dollars in safe.
- 287 5. **Reach Target:** Move the gripper to a specified target location.
- 288 6. **Slide Block:** Slide a block across the table to certain area.
- 289 7. **Stack Wine:** Put wine bottles into a shelf.
- 290 8. **Take Money:** Take dollars from safe.
- 291 9. **Take Umbrella:** Retrieve an umbrella from a stand.
- 292 10. **Pick up Cup:** Grasp and lift a cup.
- 293 11. **Unplug Charger:** Disconnect a charger from an outlet.
- 294 12. **Close Door:** Shut a door fully.
- 295 13. **Open Box:** Lift the lid of a box.
- 296 14. **Open Fridge:** Pull the fridge door open.
- 297 15. **Frame off Hanger:** Remove a frame from a hanger.
- 298 16. **Open Oven:** Open the oven door.
- 299 17. **Books on Shelf:** Put book on a shelf.
- 300 18. **Wipe Desk:** Wipe a desk surface clean using a cloth.

Table 2: Performance of HDP and HEP on 7 Tasks.

Method(Open-loop)	Mean	Reach Target	Pick Up Cup	Open Box	Open Drawer	Open Microwave	Open Oven	Knife on Board
HDP	74	<b>100</b>	82	90	90	26	58	72
HEP (Ours)	<b>94(+20)</b>	<b>100</b>	<b>98(+16)</b>	<b>100(+10)</b>	<b>94(+4)</b>	<b>82(+56)</b>	<b>87(+29)</b>	<b>96(+24)</b>

Table 3: Performance of Different Ablations on Various Tasks.

Method	Mean	Lamp on	Open microw.	Push 3 buttons	Push button	Open box	Insert USB
No Hierarchy	0.51	0.28	0.42	0.01	0.96	0.99	0.38
No Equi No FT	0.60	0.21	0.44	0.53	0.96	0.99	0.51
No Equi	0.70	0.41	0.53	0.67	0.98	0.99	0.64
No FT	0.78	0.75	0.56	0.73	0.98	0.99	0.68
No Stacked Voxel	0.84	0.77	0.65	0.87	0.99	0.99	0.79
<b>Complete Model</b>	<b>0.94</b>	<b>0.95</b>	<b>0.82</b>	<b>0.99</b>	<b>1.00</b>	<b>1.00</b>	<b>0.90</b>

- 301 19. **Cup in Cabinet**: Place a cup inside a cabinet.
- 302 20. **Shoe out of Box**: Remove a shoe from its box.
- 303 21. **Open Microwave**: Open a microwave door.
- 304 22. **Turn on Lamp**: Activate a lamp using its switch.
- 305 23. **Open Grill**: Lift the lid of a grill.
- 306 24. **Stack Blocks**: Stack blocks on top of each other.
- 307 25. **Stack Cups**: Arrange cups in a stacked configuration.
- 308 26. **Push 3 Buttons**: Press three buttons in a specific sequence.
- 309 27. **Plug USB in Computer**: Insert a USB device into a port.
- 310 28. **Open Drawer**: Pull a drawer open.
- 311 29. **Put Item in Drawer**: Pull a drawer open and place an object inside a drawer.
- 312 30. **Sort Shape**: Put shape in a shape sorter

## 313 G Real-World Experimental Settings

314 Our real-world experiments are conducted on a UR5e robotic arm equipped with a Robotiq 2F-85  
 315 gripper and three Intel RealSense D455 cameras as shown in [Figure 4](#). Demonstrations are collected  
 316 using a 6-DoF 3DConnexion SpaceMouse at a 10 Hz rate, logging both the visual observations (from  
 317 all three cameras) and the robot’s end-effector actions (position, orientation, and gripper states).

## 318 H Comparison With Hierarchical Diffusion Policy(HDP)

319 We also compare our policy with another hierarchical baseline, HDP [\[13\]](#), by selecting seven tasks  
 320 from the HDP paper that we evaluated. We then compare the success rates on these tasks, as shown  
 321 in [Table 2](#). Our approach achieves an absolute mean improvement of 20%, demonstrating superior  
 322 sampling efficiency.

## 323 I Full Result of Ablation Study

324 We show the full result of ablation study ([subsection U.4](#)) here at [Table 3](#).

## 325 J Voxelization Details

326 We build our voxelization function based on [\[3\]](#). The size of our voxel grid is 64\*64\*64 with  
 327 maximum 6 points within it.

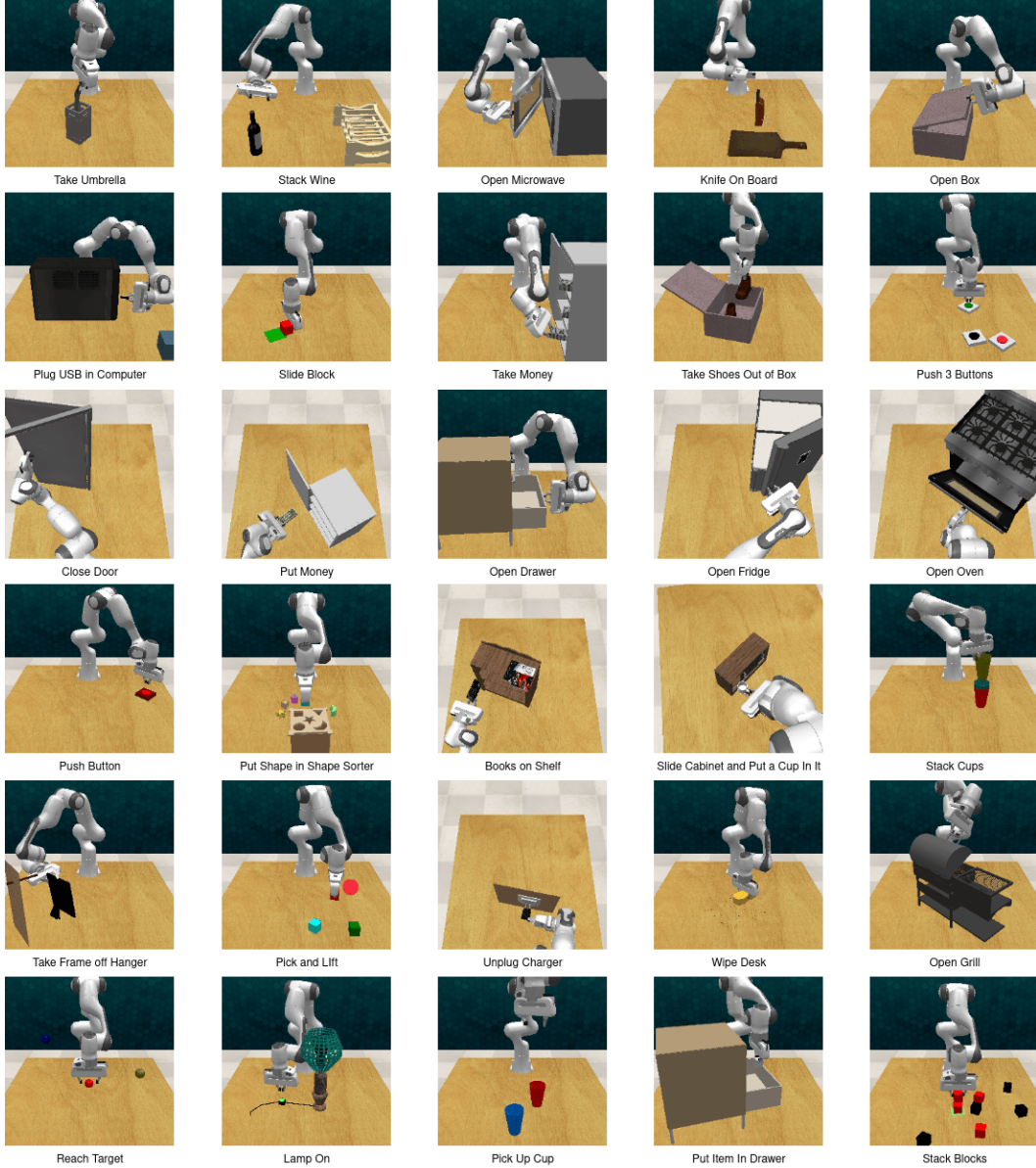


Figure 3: All simulation tasks we evaluate on

## K Equivariance Error

We conducted an experiment measuring the equivariance error specifically on  $C_4$ , a subgroup of  $SO(2)$ . This allowed us to quantify the difference between the rotated output and the output from a rotated input. The experimental results are summarized in the [Table 4](#).

## L Performance Under Human Perturbation

We conducted an extra experiment evaluating the success rate of executing a block stacking task under human perturbation mimicking a dynamic environment and include the results in [Table 5](#).

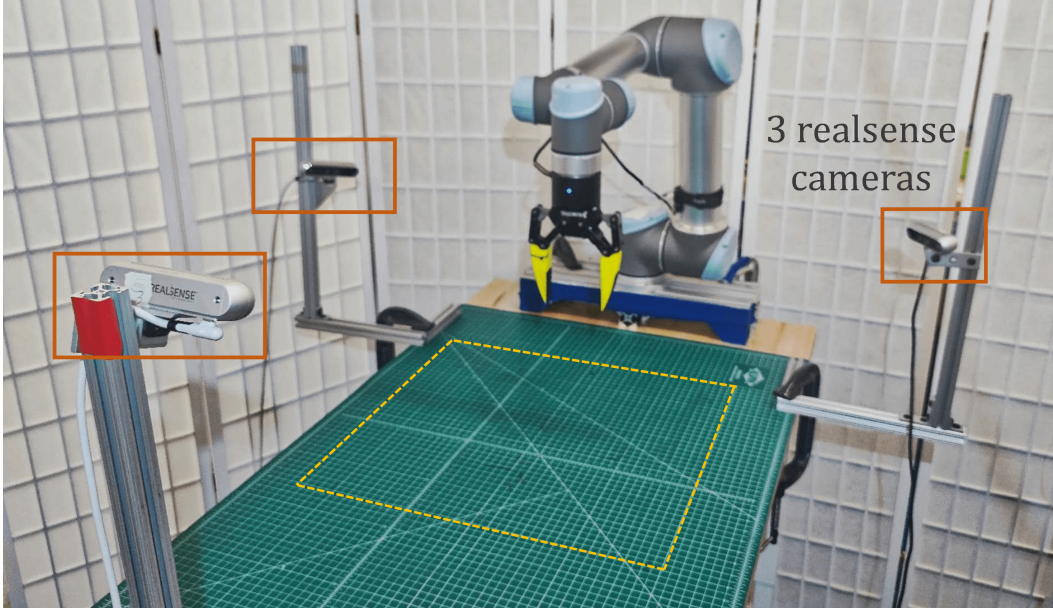


Figure 4: **Real-world Experiment Setting**

Table 4: **Equivariance Error Under Different Rotations**

Rotation	Equivariance Error
0°	0%
90°	0.013%
180°	0.006%
270°	0.009%

## 335 **M Detailed Observation and Action Spaces**

336 We define the gripper state as

$$s = (x, y, z, q, c) \in \mathbb{R}^3 \times SO(3) \times \mathbb{R},$$

337 where  $(x, y, z)$  is the end-effector position,  $q$  is the orientation, and  $c$  is the gripper aperture (open  
338 width).

339 **Observation space.** The full observation includes both visual inputs and proprioceptive history:

$$o \in O = \mathbb{R}^{n \times (3+k)} \times S^t,$$

340 where the first term represents a point cloud  $P = \{p_i : p_i = (x_i, y_i, z_i, f_i)\}$  with  $k$ -dimensional  
341 point features (e.g., RGB, segmentation mask), and the second term encodes  $t$  previous steps of the  
342 gripper state.

343 **Action space.** The action is a sequence of gripper states,

$$a = \{a_1, a_2, \dots, a_m\} \in A = S^m,$$

344 where each  $a_j = (x_j, y_j, z_j, q_j, c_j)$  specifies one control step.

345 —

## 346 **N Group Actions and Equivariance**

347 A function  $f$  is equivariant to a group  $G$  if

$$f(g \cdot x) = g \cdot f(x), \quad \forall g \in G.$$



Table 5: **Success Rate Under Human Perturbation**

Task	Success Rate
Blocks stacking	0.8

### 348 **N.1 Group Definition**

349 We consider  $G = T(3) \times SO(2)$ , where  $T(3)$  denotes 3D translations and  $SO(2)$  denotes planar  
 350 rotations around the world  $z$ -axis.

351 Let  $g = (t, R_\theta)$  with  $t = (t_x, t_y, t_z)$  and  $R_\theta \in \mathbb{R}^{2 \times 2}$  a rotation matrix.

352 —

### 353 **N.2 Action Transformation**

354 For  $a = (x, y, z, q, c)$ , define

$$\tilde{R}_\theta = \begin{bmatrix} R_\theta & 0 \\ 0 & 1 \end{bmatrix}.$$

355 The transformed action is

$$ga = (R_\theta(x + t_x, y + t_y), z + t_z, \tilde{R}_\theta q, c).$$

356 For trajectories  $a = \{a_i\}$ , apply  $g$  to each element.

357 —

### 358 **N.3 Point Cloud Transformation**

359 For a point cloud  $P = \{p_i = (x_i, y_i, z_i, f_i)\}$ ,

$$gp_i = (R_\theta(x_i + t_x, y_i + t_y), z_i + t_z, f_i).$$

360 —

### 361 **N.4 Voxel Feature Maps**

362 A voxel map  $V \in \mathbb{R}^{m \times D \times H \times W}$  can be expressed as a function

$$\mathcal{V} : \mathbb{R}^3 \rightarrow \mathbb{R}^m.$$

363 The group action is

$$(g\mathcal{V})(x, y, z) = \rho(\theta)\mathcal{V}(R_\theta^{-1}(x - t_x, y - t_y), z - t_z),$$

364 where  $\rho(\theta) \in GL(m)$  is a representation matrix that defines how features transform under  $SO(2)$ .

365 —

## 366 **O Representations of $SO(2)$**

367 We highlight common representations used in practice:

- 368 • **Trivial Representation**  $\rho_0(g)x = x$  for invariant scalars.
- 369 • **Standard Representation**  $\rho_1(g)v = R_\theta v$  for vectors in  $\mathbb{R}^2$ .
- 370 • **Regular Representation**  $\rho_{\text{reg}}$  for cyclic subgroups  $C_u$ , implemented as permutation matrices.
- 371

372 A general representation can be written as a direct sum

$$\rho = \rho_0^{n_0} \oplus \rho_1^{n_1} \oplus \rho_2^{n_2},$$

373 acting on  $\mathbb{R}^{n_0 + 2n_1 + 2n_2}$  as a block-diagonal matrix.

374 —

## P Additional Notes

- Translation symmetry  $T(3)$  is naturally handled by 3D convolutions.
- $SO(2)$  equivariance is implemented via group-equivariant layers (e.g., `escnn`).
- Variables:  $k$  = point features,  $t$  = history steps,  $m$  = trajectory length.

## Q Frame Transfer Details

The Frame Transfer operator  $\tau$  expresses observations or actions in a keypose-centered frame. Given a high-level translation  $t_{\text{high}} \in T(3)$ , we define:

$$\tau(o, t_{\text{high}}) = o - t_{\text{high}}, \quad \tau(a, t_{\text{high}}) = a - t_{\text{high}},$$

where subtraction shifts only the  $(x, y, z)$  coordinates of observations  $o$  or actions  $a$ . The inverse transformation restores world coordinates:

$$\tau^{-1}(a, t_{\text{high}}) = a + t_{\text{high}}.$$

This design ensures that relative observations and trajectories are aligned in the keypose frame, simplifying low-level learning while preserving geometric structure.

## R Stacked Voxel Encoder

A point cloud  $P$  is partitioned into voxel sets  $\{P_j\}$  on a  $H \times W \times D$  grid. For each voxel  $j$ , we compute an aggregated feature using a PointNet:

$$l(P_j) \mapsto \mathcal{V}(j_x, j_y, j_z).$$

This yields a voxel feature map  $\mathcal{V} : \mathbb{R}^3 \rightarrow \mathbb{R}^c$ . Compared to average pooling, stacked voxels preserve fine-grained point interactions inside each voxel, retaining more local shape detail.

**Proposition R.1.** *If  $l$  is  $SO(2)$ -equivariant and  $T(3)$ -invariant, then the stacked voxel encoder  $\nu : P \mapsto \mathcal{V}$  is equivariant under  $T(3) \times SO(2)$ .*

In practice,  $T(3)$ -invariance is achieved using relative coordinates within each voxel, and  $SO(2)$ -equivariance is implemented with `escnn` [11].

## S Low-level Diffusion Policy

The low-level agent  $\phi$  builds on the Equivariant Diffusion Policy [20]. Given frame-transferred observation  $o^* = \tau(o, t_{\text{high}})$  and trajectory  $a^* = \tau(a, t_{\text{high}})$ , the diffusion model predicts noise  $\varepsilon$  in the denoising process:

$$\mathcal{L}_{\text{low}} = \mathbb{E}_{k, e^k} \|\varepsilon(o^*, a^* + e^k, k) - e^k\|^2,$$

where  $k$  is the diffusion step and  $e^k$  is Gaussian noise. Equivariance is enforced by requiring

$$\varepsilon(g o, g a^k, k) = g \varepsilon(o, a^k, k), \quad g \in SO(2).$$

## T Equivariance Proofs

We formalize that HEP is equivariant under  $T(3) \times SO(2)$ .

**Proposition T.1** (Hierarchical  $SO(2)$  Equivariance). *If (i)  $\pi_{\text{high}}$  is  $SO(2)$ -equivariant, (ii)  $\pi_{\text{low}}$  is  $SO(2)$ -equivariant, and (iii)  $\tau$  commutes with  $SO(2)$  transformations, then the full policy  $\pi$  is  $SO(2)$ -equivariant.*

**Proposition T.2** (Hierarchical  $T(3)$  Equivariance). *If (i)  $\pi_{\text{high}}$  is  $T(3)$ -equivariant and (ii)  $\tau$  is  $T(3)$ -invariant, then the full policy  $\pi$  is  $T(3)$ -equivariant, regardless of whether  $\pi_{\text{low}}$  is  $T(3)$ -equivariant.*

Thus, rotating or translating the scene transforms the high-level and low-level outputs in exactly the same way, ensuring the hierarchical policy respects the underlying geometry.

## U Simulation Experiment

Table 6: **Performance of Different Models Across Various Tasks in Simulation.** Success rates (in percentages) are reported for each task. Bolded values indicate the best performance for each task, and improvements are shown in blue where applicable.

Method ( <b>Open-loop</b> )	Mean	Pick/Lift	Push Button	Knife on Board	Put Money	Reach Target	Slide Block	Stack Wine	Take Money	Take Umbrella	Pick up Cup
Ours (HEP)	<b>88(+10)</b>	<b>99(+1)</b>	<b>100(+1)</b>	<b>96(+5)</b>	98(-1)	<b>100</b>	<b>100(+2)</b>	<b>100(+7)</b>	90(-10)	<b>100(+1)</b>	<b>98(+4)</b>
Chained Diffuser	78	98	96	91	<b>99</b>	<b>100</b>	98	93	<b>100</b>	96	94
3D Diffuser Actor	56	98	99	84	88	<b>100</b>	98	90	89	99	94
Method ( <b>Open-loop</b> )		Unplug Charger	Close Door	Open Box	Open Fridge	Frame off Hanger	Open Oven	Books on Shelf	Wipe Desk	Cup in Cabinet	Shoe out of Box
Ours (HEP)		<b>99(+4)</b>	<b>90(+24)</b>	<b>100(+4)</b>	<b>83(+15)</b>	<b>93(+8)</b>	<b>87(+1)</b>	<b>99(+7)</b>	<b>77(+12)</b>	<b>76(+8)</b>	<b>90(+12)</b>
Chained Diffuser		95	76	96	68	85	86	92	65	68	78
3D Diffuser Actor		49	7	15	41	71	3	36	5	1	21
Method ( <b>Open-loop</b> )		Open Microwave	Turn on Lamp	Open Grill	Stack Blocks	Stack Cups	Push 3 Buttons	USB in Computer	Open Drawer	Put Item in Drawer	Sort Shape
Ours (HEP)		<b>82(+26)</b>	<b>95(+55)</b>	<b>99(+4)</b>	<b>54(+4)</b>	<b>32(+4)</b>	<b>99(+12)</b>	<b>90(+16)</b>	<b>94(+10)</b>	<b>95(+7)</b>	<b>22(+3)</b>
Chained Diffuser		56	40	95	10	12	86	74	84	88	10
3D Diffuser Actor		46	20	70	50	28	87	42	71	70	19
Method ( <b>Closed-loop</b> )	Mean	Turn On Lamp	Open Microwave	Push 3 Buttons	Open Drawer	Put Item in Drawer	Slide Block	Stack Wine	Take Money	Take Umbrella	Pick up Cup
Ours (HEP)	<b>79(+22)</b>	<b>60(+32)</b>	<b>64(+22)</b>	<b>37(+36)</b>	<b>95(+41)</b>	<b>76(+28)</b>	<b>95(+20)</b>	<b>89(+10)</b>	<b>94(+14)</b>	<b>90(+9)</b>	<b>93(+15)</b>
EquiDiff	57	28	42	1	54	48	75	79	80	81	78

### U.1 Experimental Settings

To evaluate our policy, we first perform experiments in simulated environments in the RL Bench [8] benchmark implemented using CoppeliaSim [14] and PyRep [7]. The simulated environments contain a 7-joint Franka Panda robot equipped with a parallel gripper, as well as four RGB-D cameras to provide the point cloud observation.

We evaluate our model on 30 RL Bench tasks, among which 20 are widely used in the prior works like [21]. The remaining 10 are challenging tasks that demand precise control, such as Lamp On, or long-horizon planning, like Push 3 Buttons. A subset of the 30 simulation tasks is shown in Figure 5. Each task is trained using 100 demonstrations, more detailed task descriptions and visualizations are provided in Appendix F.

We consider two different control settings, open-loop and closed-loop control. In closed-loop, we use each control step in the dataset as the low-level’s target, and next keyframe is used as the label for the high-level agent. In open-loop, we use the keyframe (i.e., some key actions in the entire trajectory like pick, place, etc.) defined by the prior work [17] as the target for the high-level agent, then construct the low-level target by interpolating between the consecutive keyframes. In principle, the open-loop setting requires fewer prediction steps to finish a task, while the closed-loop setting makes the policy more responsive. Thanks to the flexibility of our Frame Transfer interface, our policy can operate in both settings, while some prior works are limited in the open-loop setting.

### U.2 Baseline

We compare our method against the following baselines. **3D Diffuser Actor**: an open-loop agent that combines diffusion policies [2] with 3D scene representations. **Chained Diffuser**: an open-loop hierarchical agent that uses Act3D [4] in the high-level and diffusion policy in the low-level. **Equivariant Diffusion Policy (EquiDiff)**: an  $SO(2)$ -equivariant, closed-loop policy that applies equivariant denoising.

### U.3 Results

Table 6 presents the comparison in terms of the evaluation success rates of the last checkpoint across 100 trials.



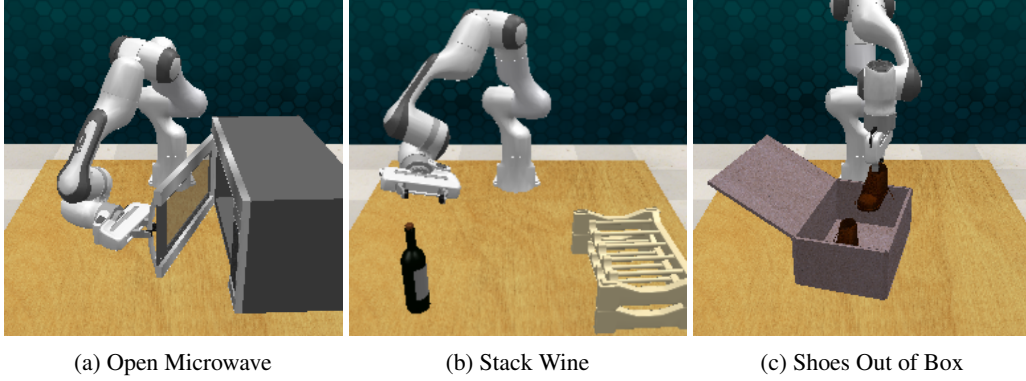


Figure 5: The Simulation Tasks from RL Bench [8]. See Appendix F for all environments.

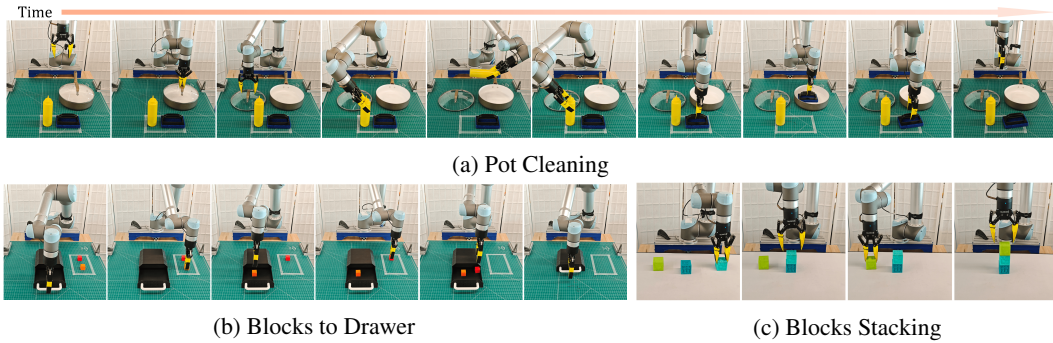


Figure 6: **Real-world Experiment Setting.** Figure 6a: Pot cleaning, the robot needs to open the pot lid, pour detergent into the pot, and clean it with a sponge. Figure 6b: Blocks to drawer, the robot needs to open the drawer, place two blocks inside, and close the drawer. Figure 6c: Blocks stacking, the robot needs to stack three blocks one by one.

**Open-loop Results:** Our model outperforms the baselines in 28 out of the 30 tasks, achieving an average absolute improvement of **10%**. The task where HEP falls short of achieving the best results is *Take Money*. Further investigation reveals that HEP achieves 98% success rate at earlier checkpoints but fails at the final checkpoint, likely due to overfitting. Tasks involving precise actions or long-horizon trajectories e.g., *Lamp-on* and *Push 3 Buttons* also exhibited consistently high success rates, demonstrating the adaptability of our method to diverse task requirements. We also compare our model with hierarchical diffusion policy [13] in Appendix H

**Closed-loop Results:** Here we consider 10 selected tasks that represent the full diversity and complexity of the complete task set. The closed-loop setting requires longer-horizon trajectories, making it harder to succeed in evaluation. Despite this, our model consistently outperforms EquiDiff across all 10 tasks, achieving an average absolute improvement of **23%**. This improvement underscores the effectiveness of HEP in handling the increased complexity of long-horizon decision-making.

#### U.4 Ablation Study

To validate the impact of our contributions, we perform an ablation study in six tasks considering the following configurations: **No Hierarchy**: removes high-level agent and uses low-level agent only. **No Equi**: same architecture but removes all equivariant structure. **No Stacked Voxel**: removes the stacked voxel encoder. **No FT**: removes the Frame Transfer interface and uses the high-level action as an additional conditioning in the low-level. **No Equi No FT**: combination of No Equi and No FT.

Table 7: Ablation Study Results.

Method	Mean
No Hierarchy	0.51
No Equi No FT	0.60
No Equi	0.70
No FT	0.78
No Stacked Voxel	0.84
<b>Complete Model</b>	<b>0.94</b>

Table 8: Performance of Different Models in the Real-World.

Task	Pot Cleaning	Blocks to Drawer	Blocks Stacking
Number of Demo	30	20	30
Chained Diffuser	0.3	0.2	0.4
Open-loop HEP (Ours)	<b>0.8</b>	<b>0.85</b>	<b>0.9</b>
Closed-loop HEP (Ours)	-	<b>0.8</b>	<b>0.9</b>

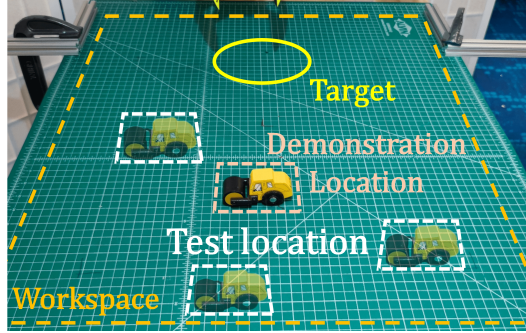


Figure 7: **One-Shot Test.** The model is trained on a single demonstration to evaluate its generalization capability.

As is shown in Table 7, removing equivariance makes the most significant negative impact on our model, reducing the mean success rate by 24%. The performance drop when removing Frame Transfer and stacked voxel encoder is 16% and 10%, respectively, demonstrating the importance of all three key pieces of our model. Moreover, the 10% performance difference between No Equi and No Equi No FT shows the potential of Frame Transfer beyond our model. See Table 3 in the Appendix for the full table.

## V Real-World Experiment

In this section, we evaluate our method on a real robot system comprised of a UR5 robot and 3 Intel Realsense [10] D455 RGBD sensors. Details on the experiment setting are given in Appendix G.

**Baseline Comparison** We experiment in three tasks as shown in Figure 6. These tasks are challenging due to their extreme long horizon (can be divided into 6 to 9 sub-tasks) and the diverse types of manipulation involved. Evaluations are conducted in 20 trials: 10 with object placements similar to the training dataset’s and 10 with unseen placements.

As shown in Table 8, our model successfully completes the tasks under open-loop control. Most failures occur due to the slight misalignment between the gripper and the object, likely caused by poor depth quality of the sensors. We further evaluate our model in a closed-loop setting, where it achieves similar performance to the open-loop version in two of the three tasks. However, in Pot Cleaning, while the agent progresses further in the task, it becomes stuck in a recurrent cleaning loop. This likely results from the lack of history information in the observations, preventing the agent from recognizing when to exit the cleaning phase. In contrast, the open-loop version follows a single keypose for cleaning, facilitating a smoother transition to the next stage.

**One-Shot Generalization** To evaluate the generalizability of our model, we perform a one-shot experiment where the model is trained to finish a pick-place task with only one demonstration. During testing, the object is placed in unseen poses, as shown in Figure 7. The results in Table 9 demonstrate the strong generalizability of our model, achieving an 80% success rate over 20 trials. For comparison, we evaluate Chained Diffuser under the same setting, but it only succeeded when the toy car was positioned exactly as in the demonstration. This result highlights the superior generalization ability of our approach, enabling robust execution of manipulation tasks from limited training data.

Table 9: **Results of One-Shot Generalization Experiment.**

Model	Success Rate
Chained Diffuser	0.05
HEP (Ours)	<b>0.80</b>

Table 10: **Results of Environmental Variation Experiment.**

Method	No Variation	Color	Color+Objects
Chained Diffuser	0.4	0	0
HEP (Ours)	<b>0.9</b>	<b>0.9</b>	<b>0.6</b>

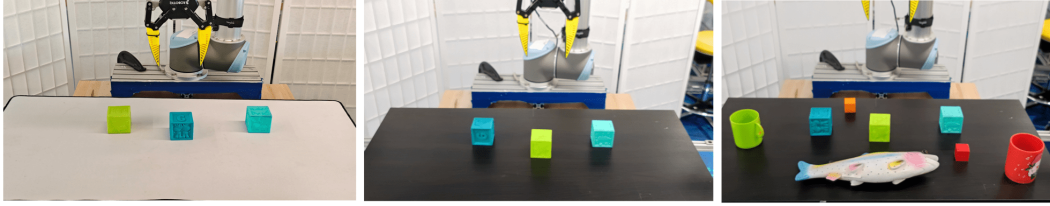


Figure 8: **Environment Variations.** Left shows the training environment. Middle and right are the test environment with variation.

490 **Robust to Environment Variations** In this experiment, we evaluate the robustness of our trained  
 491 model under environmental variations. Specifically, we introduced modifications to the Block  
 492 Stacking task during test time by changing the color of the table (Color) and additionally adding  
 493 unrelated objects as distractors (Color+Objects), as shown in Figure 8. The result is shown in  
 494 Table 10. Surprisingly, our model demonstrates exceptional adaptability, achieving 90% and 60%  
 495 success rate under those two test-time variations, whereas the baseline fails to complete the task with  
 496 those distractions.